Multistage Interconnection Network Recursive-Clos(R-Clos): Emulating the hierarchical multi-bus

Tomohiro Morimura Keisuke Iwai Hideharu Amano
Department of Computer Science
Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223-8522 Japan

Abstract

To improve practical execution performance of a multiprocessor, multi-grain parallel processing has been proposed. A multiprocessor ASCA is dedicated architecture for such multi-grain parallel processing. For an interconnection network used in ASCA, R-Clos consisting of hierarchically extended Clos networks is proposed. Using this network, a hierarchical multi-bus model can be emulated without conflicts, and the network traffic can be completely scheduled by the compiler.

The probabilistic simulation results demonstrate that the native communication performance of R-Clos is satisfactory under the traffic with locality. The required hardware is only 16% of that for Clos network when the connected PEs are 1024.

Keywords: Multistage Interconnection Network, Multi-grain Parallel Processing, Clos Network

1 Introduction

Although multiprocessor systems are widely used in recent years, it is still difficult for most users to write parallel programs to exploit their performance as possible. In order to help them, automatic parallelization compilers which transform a program in traditional sequential language like FORTRAN into parallel codes for the machine are essential. A lot of studies on automatic parallelization compilers have been reported[1][2] and some of them were commercialized. However, most of them use only loop-level parallelization techniques[3]. In order to exploit further parallelism, a multi-grain parallelizing compilation scheme has been proposed[4]. In this method, various levels of parallelism: coarse-grain parallelism (macro-data-flow computation), medium-grain parallelism (loop-level parallelism), and near-fine-grain parallelism (statements level parallelism) is detected and utilized.

Although this scheme is useful for various multiprocessors, maximum performance will be obtained on a dedicated multiprocessor architecture. For example, if a behavior of the multiprocessor is completely predictable, the compiler can schedule all tasks perfectly at the compile-time. The compiler can also remove explicit synchronization in near-fine -grain parallel processing in basic block[5]. For this purpose, we proposed a multi processor system ASCA(Advanced Scheduling oriented Computer Architecture)[8].

Due to carry out multi-grain parallel processing efficiently on ASCA, the interconnection network should be suited for the scheme. Especially communication latency among processing elements must be predicted exactly on compile-time to remove explicit synchronization in near-fine-grain parallel processing in basic block. Furthermore, since each grain has a different request for the interconnection network, the network needs to satisfy those characteristics independently.

In this paper, we propose a multistage network suited for multi-grain parallel processing, called Recursive Clos (R-Clos), which can emulate the hierarchical multi-bus model.

2 Networks for the multi-grain parallel processing

2.1 Requirements for networks

For multi-grain parallel processing, cluster structure for a certain number of processors is required. Each macro task is assigned into a cluster, and the loop level parallel processing or near fine grain parallel processing is done inside the cluster. Since the optimal number of processors in a cluster is various with the size of a macro task, a network which can form a flexible cluster structure is desirable. The coarse grain parallelism is utilized processing between clusters.

Data transfer manner for multi-grain parallel processing is as follows:

- **near-fine grain**:
  Most of data transfer for near-fine-grain parallel processing is caused by the access of variables between statements. Thus, small size data is transferred frequently.
- **medium grain**:
  Medium-grain parallel processing corresponds to the loop level (Do-Across loop or Do-all loop) parallel processing. In case of Do-All loop, data transfer is required only when checking the condition of exiting the loop.

- **coarse grain**:
  This level mainly requires a large size data transfer between macro tasks. Although the data size is large, this communication does not occur frequently. Macro tasks are assigned to processor clusters at run time, and synchronization is required for every data transfer.

From these properties, requirements for the network are summarized as follows.

- The network structure
  - must form a cluster which enables closed partitioning, and
  - should be a simple for scheduling the data transfer.
- Communication inside the cluster requires
  - a transfer with fixed and small latency, but
  - the data size is fixed.
- Communication between clusters requires
  - a transfer with a large peak bandwidth, but
  - the latency is not critical.

In order to satisfy the above requirement, for inner cluster communication network should have strong transfer ability and characteristics of an equal distance network that enable transfer latency to be predicted and provide short time transfer, while for outer cluster communication it should provide large peak bandwidth of data transfer.

3 Multi-stage Interconnection Network R-Clos

3.1 Structure of R-Clos

3.1.1 Clos network

Clos network is a multistage interconnection network consisting of three stages: distributor, exchanger and concentrator, as shown in Fig.1. The network capacity of Clos is based on relationship between the number of inputs/outputs number of distributor/concentrator and number of elements of exchanger. Here, let the number of inputs/outputs of distributor and concentrator be the same (n). It becomes a nonblocking network when the number of exchanger elements (m) is larger than or equal to $(2n-1)$. If m is larger than or equal to n, it is a rearrangeable network which can be nonblocking with scheduling. Fig.1 shows such a rearrangeable structure. Otherwise it becomes a blocking network like traditional networks used in multiprocessors. Although high degree of bandwidth and schedulability of Clos network are suitable for our goal, the following problems remain:

- For a large size system, a large number of switching elements is required.
- Since it is a flat network, the latency is the same from all inputs to outputs. It is not suitable for cluster structure.

We introduce a hierarchical structure into Clos network and proposed a new network called R-Clos network.

3.1.2 R-Clos network

R-Clos is an extended Clos network[6] with hierarchical structure by introducing additional exchanger stages. As shown in Fig.2, a certain size of Clos network forms an elementary rank-1 cluster, and the exchanger of the rank-1 cluster is expanded so as to connect another rank-1 clusters. Each rank-n cluster is recursively formed by k rank-$(n-1)$ cluster.

If the number of processing elements which must be connected with R-Clos is P and the size (the number of inputs/outputs) of switching element is k, the rank R is fixed with the following expression.

$$R = \log_k P - 1$$

The number of additional stages S is as follows:

$$S = (R-1)(R \geq 2).$$

3.2 Definitions of R-Clos

Definition 1 [Definitions of Clos used as a basis]

Here, unidirectional Clos network in which all switching elements consist of $k \times k$ crossbar is used as the basis. The input stage, middle stage, and output stage are called distributor, exchanger, and concentrator respectively. A sequential number is assigned into k a switching element in every stage, and a switching element assigned with number j in distributor, exchanger, and concentrator are called D_j, $E_{1,j}$ and C_j respectively. The connection rule is as follows:

- *The output port m of D_j is connected to the input port j of $E_{1,m}$.*
- *The output port m of $E_{1,j}$ is connected to the input port j of C_m.*

This network is proved to be rearrangeable according to [6].

Definition 2 [Definitions of R-Clos$(i,k)(0<i)$]

Here, R-Clos(i,k) represents a rank-i R-Clos with radix-k clustering. First, R-Clos(0,k) is a simple $k \times k$ crossbar, and R-Clos(1,k) is the base Clos network shown the above. R-Clos(i,k) $(i \geq 2)$ is formed with

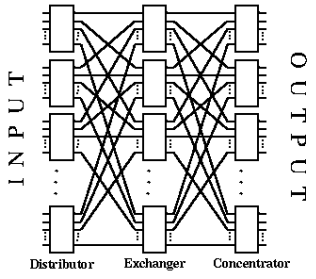


Figure 1: *Clos* Network

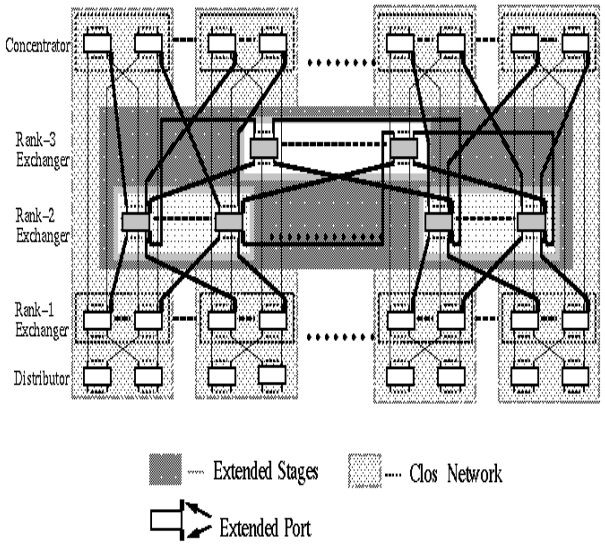


Figure 2: *Structure of rank 3 R-Clos network*

k extended R-Clos(i − 1,k) and additional exchanger stage as follows:

1. *Form an additional exchanger (rank-i exchanger) with k $k \times k$ (input port 0 to $k-1$, output port 0 to $k-1$) switching elements ($E_{i,m}, k > m > 0$).*
2. *Assign a sequential number $j(k > j > 0)$ into k R-Clos(i − 1,k) networks. An R-Clos(i − 1,k) which is assigned with the number j is called R-Clos$(i-1,k)_j$.*
3. *Add an extended output port (labeled k) into every $E_{i-1,m}$ in every R-Clos(i − 1,k).*
4. *Connect every output port k of $E_{i-1,m}$ in the R-Clos$(i-1,k)_j$ to input port j of $E_{i,m}$.*
5. *Add an extended input port (labeled k) into every $E_{i-1,m}(if\ i \neq 2)$ $(C_m, i = 2)$ in every R-Clos(i − 1,k).*
6. *Connect the output port j of $E_{i,m}$ to every input port k of the $E_{i-1,m}(if\ i \neq 2)$ $(C_m, i = 2)$ in the R-Clos$(i-1,k)_j$.*

By this connection, R-Clos(i,k) consisting of R-Clos$(i-1,k)_0$, R-Clos$(i-1,k)_1$, ... R-Clos$(i-1,k)_{k-1}$ is formed. Since it is a hierarchical network, each cluster is labeled by the tag of R-Clos(i,k),R-Clos$(i-1,k)_{x_{i-1}}$, ... R-Clos$(1,k)_{x_1}$

The number of elements of rank-n exchanger stage $N(E_n)$is given by the following equation.

$$N(E_n) = \left(\frac{P}{k}\right)\left(\frac{1}{k}\right)^n \qquad (n < R) \tag{1}$$

We call an *extended output/input port* (port k) of switching element in the rank-1 exchanger, rank-1 concentrator, and rank-i exchanger *extended port*, and a link connected to the *extended port* is called the *extended link*. Other ports (port 0 to $k-1$) are called *basic ports*.

3.3 Routing Scheme

In order to establish the routing scheme, input/output links of the network must be labeled. Here, the number assigned into each cluster of R-Clos in Definition2 and number assigned into each element in Definition1 are directly used.

That is, an s_0 input port of D_{s_1} which belongs to R-Clos(R,k), R-Clos$(R-1,k)_{x_{r-1}}$,... R-Clos$(R-1,k)_{x_1}$ is labeled as: $x_{r-1}x_{r-2}\ldots x_1 s_1 s_0$, where $x_{r-1}, x_{r-2}\ldots x_1, s_1, s_0$ are radix-k number.

Similarly, an d_0 output port of C_{s_1} which belongs to R-Clos(R,k), R-Clos$(R-1,k)_{x_{r-1}}$,... R-Clos$(R-1,k)_{x_1}$ is labeled as: $x_{r-1}x_{r-2}\ldots x_1 d_1 d_0$, where $x_{r-1}, x_{r-2}\ldots x_1, d_1, d_0$ are radix-k number.

Here, a packet whose destination port is represented as $D = d_m d_{m-1} d_{m-2}\cdots d_1 d_0 (m = R-1)$. Similarly, the source port label is $S = s_m s_{m-1} s_{m-2}\cdots s_1 s_0 (m = R-1)$.

1. The case when the source and destination is in the same Rank-1 cluster ($d_m d_{m-1} d_{m-2}\cdots d_2 = s_m s_{m-1} s_{m-2}\cdots s_2$):

   In this case, packets can be routed only through *basic port*s, and the triplet Tag $T(t_0,t_1,t_2)$ is required for each packet header. According to Definition1, lower 2 digits of the label are exchanged when a packet passes any link between stages. Therefore, for destination D, the following simple tag is used and applied in order:

   $$T(\ *\ ,\ d_1\ ,\ d_0\ ) \quad (* = \forall y \mid y = 0,1,\cdots,k-1) \tag{2}$$

   That is, in the first stage (distributor), packet can select any output port.

2. The case when the source and destination is in the different Rank-1 cluster ($d_m d_{m-1} d_{m-2}\cdots d_2 \neq s_m s_{m-1} s_{m-2}\cdots s_2$):

   In this case, *extended port*s must be used. While only lower two digits of the label are exchanged by using *basic port*s, upper digits can be changed by using links connected to *extended port*s. Extended ports must be used until the source port and destination port is in the same hierarchy, that is, ($d_m d_{m-1} d_{m-2}\cdots d_{r+2} = s_m s_{m-1} s_{m-2}\cdots s_{r+2}$): By passing *extended port*s r times, a packet arrives at rank-$(r+1)$ exchanger and then selects a *basic port* to the concentrator of lower ranks.

   Thus, routing tags T is obtained as follows:

   $$T(\ d_1, \underbrace{k,\ldots,k}_{r}, \underbrace{d_{r+1},\ d_r,\ \ldots,\ d_2}_{r},\ d_0) \tag{3}$$

   Note that k represents the packet is routed to the *extended port* (output) of the switching element. There is no flexibility packet transfer between ranks, and a routing path is statically fixed.

4 Scheduling Model for R-Clos

As described in Section 2, the network must be scheduled by the compiler easily. Although the structure of R-Clos network is simple, it is still hard to be used as a target of the scheduling directly. Therefore we consider a scheduling model called the hierarchical multi-bus model, which can be easily scheduled by the compiler and easily emulated by R-Clos.

4.1 Hierarchical Multi-bus Model

As a target of scheduling, a crossbar which provides individual bus for each PE is one of the best candidates. As shown in Fig.3, hierarchical multi-bus model is an extension of the crossbar. A certain number of PEs connected with a crossbar forms a level-0 cluster, and a joint bus attached to each crossbar is connected to another crossbar to form level-1 cluster. For inner cluster communication, a bus can be assigned into PE like a crossbar. For inter-cluster communication, a PE which represents the cluster can use a bus in the upper level. This restriction is simple and compiler can easily schedule the data transfer without any conflict.

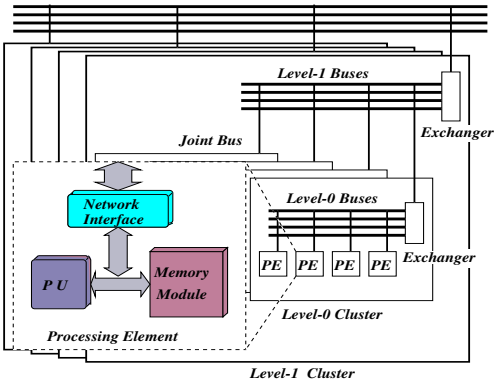


Figure 3: Hierarchical Multi-bus Model

4.2 Emulation of Hierarchical Multi-bus Model

4.2.1 Direct emulation

Next, we show the method for emulating the hierarchical multi-bus model on the R-Clos without conflict.

First, the following theorem is introduced:

Theorem 1 [Non-blocking Emulation]

R-Clos network can emulate a hierarchical multi-bus model directly without conflicts except in the rank-1 exchanger.

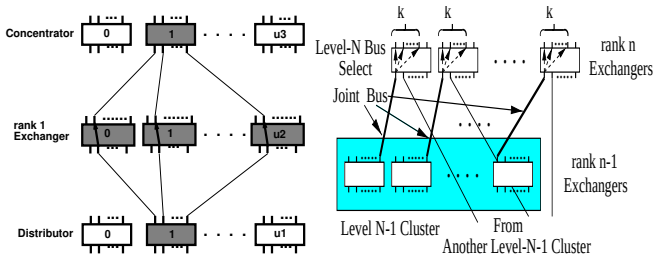
The communication of the hierarchical multi-bus is difference between level-0 buses and level-N buses. Therefore, we divide the above theorem into two parts.

Lemma1 [Non-blocking emulation on level-0 buses]

R-Clos network can emulate level-0 buses of a hierarchical multi-bus model directly without conflicts when there is no upper level communication.

In level-0 cluster of the model, the same number of buses as cluster components (k) can be used.

Therefore, we show that there are k paths from source cluster to destination cluster which can construct independently and simultaneously. Secondly, we shows that these k paths do not interfere with construction of another level buses. In rank-1 R-Clos, dedicated paths can be formed as shown in Fig.4(a), and conflict free emulation is possible.



(a) Level-0 Buses (b) Level-N Buses

Figure 4: Emulating the hierarchical multi-bus

The proof is shown below.

proof 1 [Proof for Lemma 1]

Rank-1 cluster of the R-Clos is a rearrangeable Clos network whose size of switching element corresponds to k.

Here, routing tag

$$T_0 = (*, d_1, d_0) \quad (* = \forall y | y = 0, 1, 2, \cdots, k-1)$$

is used by Equation2. Any conflicts among level-0 communications are not occurred at distributor because of having liberty of selecting any output port (given by Equation2). If it is assumed that the packet selects t_0, it is sent to the t_0-th rank-1 exchanger. Since t_0 can be any integer value from 0 to $k-1$, k paths of level-0 communication exist. Since at each rank-1 exchanger, a packet from d_1 rank-0 cluster is routed for d_1th output port, there are no conflicts between level-0 communications at each rank-1 exchanger. Thus, k level-0 communication paths on R-Clos network can be established independently. ∎

In the hierarchical multi-bus model, only one level-N communication is permitted from each level-$(N-1)$ cluster. Therefore, the following lemma is easily introduced:

Lemma2 [Non-blocking emulation on level-N buses]

R-Clos network can emulate level-N ($N > 1$) buses of a

hierarchical multi-bus model directly without conflicts.

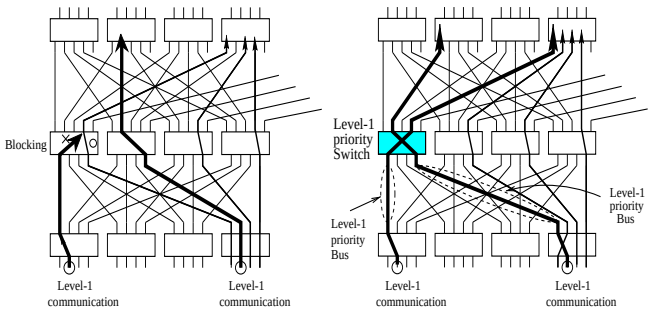
proof 2 [Proof for Lemma2]

According to Equation2, selecting and acquiring the level-N bus in the hierarchical bus model is equal to selecting the output port of rank-N exchanger of R-Clos. Since there are k output ports at each rank-N exchanger, k level-N buses can be emulated without conflicts. The communication to node belongs to further upper level uses the dedicated extend link *of the exchanger. Thus, communication for upper level cluster does not conflict.* ∎

This situation is illustrated in Fig.4(b). Note that a joint bus is emulated with all *extended output port* at each rank exchanger.

4.2.2 Introducing the priority switching element

When the routing algorithm shown in the previous section is directly used, conflicts only occur in between the communication for level-0 buses and level-1 buses as shown in Fig.5(a).



(a) Conflict between Level-0 Bus and Level-1 Bus

(b) Priority switching element

Figure 5: The solution of the conflicts

In order to avoid it, we use one of rank-1 exchangers as **the priority switching element** which works as follows:

Condition 1 [priority switching element]

[1] Level-1 communications must use this priority switching element.

[2] Level-0 communications may use this element only when any other switching element in the exchanger has been already occupied.

[3] Level-N ($N > 1$) communications may use this element except when it is occupied by level-0 communications.

Using this mechanism, the following lemma can be introduced:

Lemma3 [Non-blocking emulation of level-0 buses with level-1 communication]

Conflicts free emulation of the level-0 buses can be done on the R-Clos using the priority switching element.

proof 3 [Proof for Lemma 3]

An arbitrary switching element of the rank-1 exchanger is used as the priority switching element. From the Lemma1, conflict only occurs with level-1 buses communications and level-0 buses communication. Using the priority switching element, all packets for level-1 buses use the dedicated path to the priority switching element. Since the priority switching provides paths to every concentrator element, there is no conflict for emulating level-1 buses. Since the rest of paths can be used for emulating level-0 buses, packets for emulating level-0 buses do not conflict each other as shown in the Lemma1.

∎

This situation is shown in Fig.5(b). Since packets for level-1 bus communication are gathered in the priority switching element, the conflict shown in Fig.5(a) is resolved.

5 Evaluation of performance

5.1 Evaluation Scheme

Here, the pass through ratio and latency are analyzed with a simple simulation. When conflict occurs, one of conflicting packets are randomly selected and routed its destination. Another packets are dropped and inserted again. For estimating the latency, it is assumed that two clocks are required to pass a stage.

Following two types of traffic are assumed.

- Random traffic (16 PEs, 64 PEs, and 256 PEs)
  All packets are transferred to completely random destinations.
- Localized traffic (256 PEs)
  A certain packets are transferred randomly only in the rank-0 cluster. The rest of packets are transferred randomly to all destinations.

5.2 Results of Simulations

Random traffic simulation: Pass through ratio against issued access rate is shown in Fig.6.

R-Clos with 16PEs is just a simple Clos network. Therefore, result with 16PEs shows the performance of traditional Clos network and inner rank-1 cluster transfer capacity. Pass through ratio of 16PEs is saturated up to 55%. Considering that the pass through ratio of crossbar, Non-Blocking network, is saturated up to 65%, the transfer capacity of rank-1 cluster is satisfactory.

In 64PEs, and 256 PEs, the pass through ratio is degraded rapidly. This comes from hierarchical structure of R-Clos network.

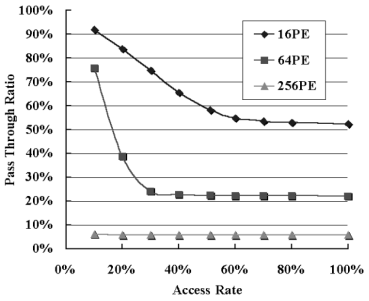
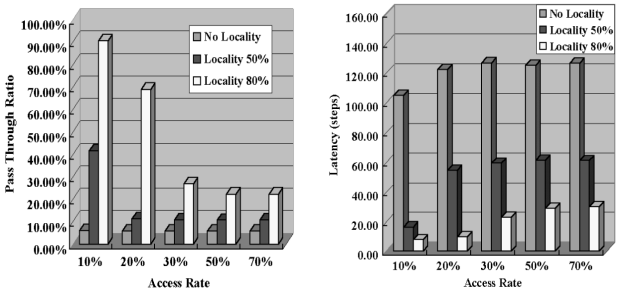


Figure 6: Pass through ratio against issued access rate under random traffic

Localized traffic: Pass through ratio when 50% and 80% of traffic is in rank-0 cluster is shown in Fig.7(a). In both cases, the pass through ratio is much improved compared with the case of random traffic. Result of latency are shown in Fig.7(b). Latency of



(a) Pass Through ratio against issued access rate

(b) Latency against issued access rate

Figure 7: The results Under localized traffic,PE = 256

traffic without locality are also shown. The latency of 50% localized traffic is about a half of non-localized traffic when the access issued rate are more than 20%. In regard to the latency of 80% localized traffic, the latecy is much reduced, and 1/6 of non-localized traffic.

When multi-grain parallel processing is performed, a large part of traffic is done between nodes in the cluster. This simulation results demonstrate that R-Clos provides enough native performance for the traffic with locality.

5.3 Estimation of hardware cost

First, we consider hardware cost of R-Clos. Let P be the number of connected nodes, that is, the number of inputs/outputs of the network. Let k be the number of inputs/outputs of the switches in basic stage(within Clos network).

The number of crosspoints of R-Clos is given by the following equation.

$$\begin{aligned} CP_{rclos} &= Pk + \frac{k(Pk+P-2k^2)}{k-1} + P(k+1) \\ &= \frac{3Pk^2 - P - 2k^3}{k-1} \end{aligned} \quad (4)$$

Table1 shows the crosspoints of R-Clos network, Clos network and crossbar calculated by the above equations.

Table 1: Number of crosspoints ($k = 4$)

PE	CP_{rclos}	CP_{clos}	CP_{cross}	CP_{rclos}/CP_{clos}
16	192	192	256	100%
64	960	1536	4096	62%
256	3968	12248	65536	32%
1024	16000	98304	1048576	16%

Table 1 demonstrates that R-Clos requires much less hardware than corresponding size of Clos network and a simple crossbar. For a larger system, the ratio becomes great.

6 Conclusion

R-Clos network consisting of hierarchically extended Clos networks is proposed for multi-grain parallel processing scheme. Using this network, a hierarchical multi-bus model can be emulated without conflicts, and the network traffic can be completely scheduled by the compiler.

The probabilistic simulation results demonstrate that the native communication performance of R-Clos is satisfactory under the traffic with locality. The required hardware is only 16% of that for Clos network when the connected PEs are 1024.

References

[1] Constantine D.Polychronopoulos, Milind B. Girkar, Mohammad R. Haghighat, Chia L. Lee, Bruce P. Leung, Dale A. Schouten: *"Parafrase-2: An Environment for Parallelizing , Partitioning, Synchronizing, and Scheduling Programs on Multiprocessor"*, In *Proc. of the International Conference on Parallel Processing*, pp.II39-48, August 1989.

[2] M. W. Hall, J. M. Anderson, S. P. Amarasinghe, B. R. Murphy, S. W. Liao, E. Bugnion, M. S. Lam: *"Maximizing Multiprocessor Performance with the SUIF Compiler"*, *IEEE Computer*, pp.84-89, December 1996.

[3] U. Banerjee: *"Loop Parallelization"*, Kluwer Academic Pub, 1994.

[4] M. Okamoto, K. Yamasita, H. Kasahara, S. Narita: *"Hierarchical Macro-Dataflow Computation Scheme on a Multiprocessor System OSCAR"*, In *Proc. of IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, pp.44-49, May 1995.

[5] W. Ogata, K. Fujimoto, M. Oota, H. Kasahara: *"Compilation Scheme for Near Fine Grain Parallel Processing on a Multiprocessor System without Explicit Synchronization'*, In *Proc. of IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, pp. 327-332, May 1995.

[6] Clos. C: *"A study of non-blocking switching networks"*, Bell system Tech.J. 32, pp.406-424, Mar 1953.

[7] H. Kasahara, S. Narita, S. Hasimoto: *"OSCAR's Architecture"*, Trans. IEICE Vol.J71-D-I, No. 8. August 1988.

[8] K. Iwai, T. Morimura, T. Fujiwara, K. Sakamoto, T. Kawaguti, K. Kimura, H. Amano, H. Kasahara: *"AN INTERCONNECTION NETWORK OF ASCA:A MULTIPROCESSOR FOR MULTI-GRAIN PARALLEL PROCESSING"*, In*Proc. of IASTED International Conference APPLIED INFORMATICS AI'98*, Feb 1998.